Adaptive Motion Planning for a Collaborative Robot Based on Prediction Uncertainty to Enhance Human Safety and Work Efficiency

Akira Kanazawa, *Member, IEEE*, Jun Kinugawa, *Member, IEEE*, and Kazuhiro Kosuge, *Fellow, IEEE*

***Abstract*—Industrial robots are expected to share the same workspace with human workers and work in cooperation with humans to improve the productivity and maintain the quality of products. In this situation, the worker's safety and work-time efficiency must be enhanced simultaneously. In this paper, we extend a task scheduling system proposed in the previous work by installing an online trajectory generation system. On the basis of the probabilistic prediction of the worker's motion and the receding horizon scheme for the trajectory planning, the proposed motion planning system calculates an optimal trajectory that realizes collision avoidance and the reduction of waste time simultaneously. Moreover, the proposed system plans the robot's trajectory adaptively based on updated predictions and its uncertainty to deal not only with the regular behavior of workers but also with their irregular behavior. We apply the proposed system to an assembly process where a two-link planar manipulator supports a worker by delivering parts and tools. After implementing the proposed system, we experimentally evaluate the effectiveness of the adaptive motion planning system.**

***Index Terms*—Adaptive control, collision avoidance, cognitive human–robot interaction, industrial robot, learning and adaptive systems.**

I. Introduction

MANY industrial robots have been used in various fields to improve the productivity while maintaining or even improving the product quality. However, in many processes, complete automation remains difficult to achieve. In reality, many complex tasks that cannot be accomplished by robots are still performed by skilled human workers. In contrast, systems in which robots work in cooperation with human workers are becoming common in several factories. In such systems, human workers remain in charge of situation assessments and precision tasks that are difficult for robots to perform. However, robots perform simple repetitive tasks and handle heavy parts instead of human workers. By exploiting their given strengths in a collaborative manner, an efficient system can be realized.

With the amendment of the International Organization for Standardization (ISO) for operating industrial robots in factories, it has become possible for robots now to share the same workspace with humans. Industrial robots with over 80 W of power are able to operate without fencing provided that appropriate risk assessment is done. This approach is expected to reduce the costs related to the securing of the workspace, to save space on the factory floor, and to improve the flexibility of the production line. In response to such changes, major industrial robot makers such as KUKA, ABB, FANUC, and Rethink Robotics are working to develop collaborative robots [1]. Consequently, the development of a human–robot collaboration system has become a key issue in the field of factory automation.

In previous work, we proposed an adaptive task scheduling system for collaborative robots [2]. The proposed system learned worker's motion patterns by combining probabilistic models with an online algorithm to predict the worker's future motion and adaptively schedule the robot's task. By explicitly considering the temporal requirement, this system decreased the time wasted by delaying the robot's task. However, the worker's safety was not sufficiently considered yet in this system. Although this system is based on a statistical prediction of the worker's motion, workers do not necessarily act in a consistent manner. For example, the working position or the way the worker moves in the workspace may change in an irregular way. When the system is implemented in actual processes, it must ensure the worker's safety while the robot continues with its task.

To address this issue, we focus herein on a motion planning system for collaborative robots working in assembly processes. In such processes, workers repeat the same work schedule according to a process sheet known *a priori*. We thus propose an adaptive motion planning system that uses the features of the assembly processes to avoid robot–worker collisions and improves the work efficiency simultaneously. Fig. 1 shows the configuration of the proposed motion planning system, which is composed essentially of a worker's motion predictor and an online trajectory generator. We use the online learning and prediction method

Manuscript received September 4, 2018; accepted March 31, 2019. Date of publication June 3, 2019; date of current version August 1, 2019. This paper was recommended for publication by Associate Editor S. Calinon and Editor A. Billard upon evaluation of the reviewers' comments. *(Corresponding author: Akira Kanazawa.)*

A. Kanazawa is with the Hitachi Ltd., Hitachinaka-shi 312-0034, Japan (e-mail: akira.kanazawa.qg@hitachi.com).

J. Kinugawa and K. Kosuge are with the Department of Robotics, Tohoku University, Sendai-shi 980-8579, Japan (e-mail: kinugawa@irs.mech.tohoku.ac.jp; kosuge@irs.mech.tohoku.ac.jp).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author. This video presents results of trajectory plannning for one regular motion pattern and four irregular motion patterns.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2019.2911800

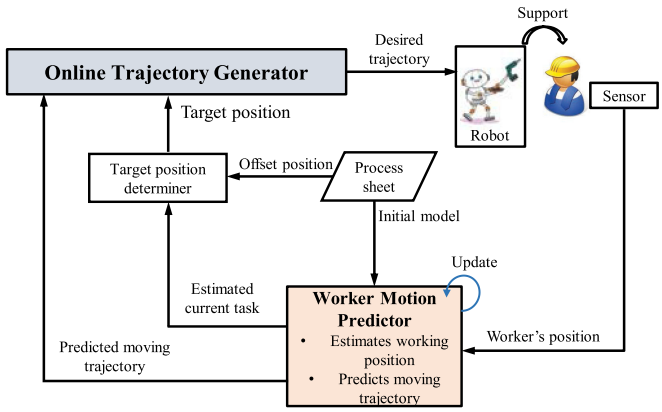


Fig. 1. Configuration of the proposed adaptive motion planning system.

proposed in [2] to predict the worker's motion. In addition, we install an online trajectory planning method proposed herein to the online trajectory generator. On the basis of the probabilistic prediction of the worker's motion by the worker motion predictor, the proposed system calculates a trajectory that enhances the human's safety and the work-time efficiency simultaneously.

This paper makes the following three contributions.

1) The proposed trajectory generator makes effective use of the predicted worker's motion. By planning the robot trajectory in the expanded temporal space that includes future worker states and its uncertainty, the proposed method explicitly incorporates temporal requirements and collision risk into a trajectory optimization problem. This optimized trajectory satisfies multiobjective requirements, including collision avoidance, smoothness, and reduction of the waste time.
2) The proposed system is automated and flexible by installing incremental update algorithms. Once the predefined parameters are determined, the system judges the situation and decides which task to do. In addition, the system deals appropriately with irregularities in the worker's motion based on outputs of the worker's motion predictor.
3) The experimental verification was done in an environment similar to an actual assembly process. We verified the effectiveness of the proposed system for both regular and irregular cases of the worker's motion and for several workers.

The rest of this paper is organized as follows. Section II summarizes related works about human collaborative robot systems for assembly processes, human motion modeling, and trajectory planning. Section III reviews the online learning and prediction method using probabilistic models. Section IV explains the details of the online trajectory planning method, which is based on the trajectory optimization using the receding horizon scheme. Section V describes the configuration of the adaptive motion planning system when applied to an actual manipulator. Section VI discusses experiments that were conducted to evaluate the proposed system, and finally, Section VII concludes this paper.

II. Related Works

A. Human–Robot Collaborative Systems in Assembly Scenario

The biggest issue when introducing a collaborative robot system into actual processes is to ensure the worker's safety. Many studies have focused on the safety issues of human–robot collaboration systems in assembly processes. For example, against the backdrop of robot safety standards, Fryman *et al.* summarized the basic types of the collaborative system in operation with the development of industrial robots [3]. To avoid collisions in a collaborative workspace, Pedrocchi *et al.* proposed a safe network of unsafe devices where manipulators are regarded as nodes in the network system [4]. Changliu *et al.* proposed a control method based on safety measures for industrial robots working in a human-related environment [5]. Lasota *et al.* proposed a platform for safely running a robot system in real time based on the measured worker data [6]. Zanchettin *et al.* proposed a kinematic control strategy based on metrics to evaluate the safety of the human–robot collaborative manufacturing [7].

In actual processes, the collaborative robot is also expected to produce value to offset the initial cost and the running cost. Therefore, improving the work efficiency is one of the most important benefits of such collaborative robots. Collaborative robots are thus required not only to operate safely in the workspace that it shares with human workers but also to help to improve the productivity. Several studies have focused on improving the work efficiency; for example, Wilcox *et al.* proposed an algorithm that optimizes task scheduling and control for robots in the assembly manufacturing [8]. In other work, Hawkins *et al.* proposed a wait-sensitive task planning method based on the predicting human action in a probabilistic manner [9]. Baizid *et al.* proposed a time scheduling and optimization approach based on a genetic algorithm, which focused on industrial robotized tasks [10].

Several studies simultaneously assured the worker's safety and the continuity of the robot's task. For example, Hyne *et al.* proposed a dynamic trajectory planning method [11]. Modeling a region with a high probability of danger, they considered the avoidance of collision and the consistency of the robot motion as an optimization problem. Ceriani *et al.* proposed an algorithm to dynamically generate an avoidance trajectory [12]. The proposed method constructs a danger field that quantifies the risk of robot–worker collision and selects the degrees of freedom for the manipulator with a redundancy joint.

Although these studies manage to plan safe the robot's motion while maintaining the continuity of the robot's original task, they did not sufficiently consider the work-time efficiency. Since temporal requirements and constraints are not taken into account in the planning strategy, the waste time due to a delay of robot's motion or change of the worker's movements may occur. To incorporate temporal requirements into motion planning, it is necessary to predict the behavior of workers. However, the workers do not necessarily take the same action as before, and the prediction of their behavior becomes uncertain. The irregular behavior of the worker makes the motion planning of the robot difficult and can become a factor of increasing the collision

risk. In this work, we address the problem of how to simultaneously enhance both worker's safety and work-time efficiency by considering the prediction uncertainty.

B. Probabilistic Modeling of Human Behaviors

In this paper, we consider predicting the human's motion in order to effectively plan the robot's tasks. In the automobile assembly process, a worker is presumed to repeat the same tasks while following a similar motion trajectory in every cycle. To take advantage of this feature, we use a statistical approach for modeling the human motion trajectory. Compared to methods considering a current human dynamical state, statistical approaches can predict one's long-term movement with high precision. In addition, repeated human movements have variations and gradually changing due to a work delay, worker's fatigue, and adaptation to one's task. Therefore, it is considered that a probabilistic model that can express the uncertainty of prediction is effective. Furthermore, it is desirable to be able to update the model parameters adaptively according to the change of the worker's movement.

One effective approach of the stochastic modeling is to utilize graphical models such as the hidden Markov model (HMM). In particular, several methods have been proposed that can adaptively modify the models based on newly obtained data. Vasquez *et al.* combined an instantaneous topological mapping algorithm with an HMM and proposed a method for predicting a human motion trajectory online [13]. Morris *et al.* conducted trajectory prediction and anomaly detection with a spatial and temporal vehicle motion model adaptively based on new image data obtained from the surveillance system [14]. Dawood *et al.* presented an approach for the incremental learning of human motion patterns using the HMM integrated with the topological Gaussian adaptive resonance theory (TGART) [15]. Although the HMM can stochastically encode spatial and temporal features simultaneously, it is difficult to decode the temporal feature in detail since the trajectories are discretized and abstracted. To explicitly incorporate temporal features into the model, modeling methods using explicit-duration HMM [16] and autoregressive HMM [17] are proposed. However, it is difficult to extend them to an online algorithm because the learning of model parameters does not converge well unless the structure of the graphical model is set in advance.

Another effective approach is to utilize a nonlinear regression method for modeling the human dynamical system. Movement primitives (MP) [18] is a popular approach for learning the dynamical system from the obtained dataset. Paraschos *et al.* extended this MP concept with the probabilistic formulation [19]. They presented probabilistic movement primitives (ProMP) that expresses a dynamical system as mixture probabilistic distributions. Similar to such ProMP concept, Gaussian mixture regression (GMR) is used for modeling and predicting the human movement trajectory [20], [21]. In the same way, as a model that utilizes probabilistic distribution, Gaussian process dynamical model (GPDM) is an effective way of modeling the human dynamical system stochastically [22]. Similarly, many approaches based on the Gaussian process regression (GPR) have been proposed [23], [24].

In the proposed system, we adopt a modeling method based on the GMR for the following advantages: 1) it can easily be extended to online modeling with incremental expectation-maximization (EM) algorithm; and 2) it can stochastically model an uncertain trajectory and predict a multidimensional output. Although GPR-based approaches can also model human trajectories and express its uncertainty with high precision, it needs to keep past datasets for prediction, which leads to an increase in calculation cost and is difficult to use as the online algorithm. In addition, the GPR is difficult to calculate a covariance between elements of the multidimensional output. The GMR retrieves the expected mean and variance of the data, while GPR retrieves the uncertainty of the mean estimate. Despite these differences, the GMR can predict a reasonable output for the outlier output based on the trend of the current model. Because small anomalies occur frequently in the actual assembly process, it is desirable to make reasonable predictions for small changes in the worker's behavior. Emphasis on the calculation cost of the online algorithm and the prediction flexibility against outlier inputs, the GMR is used for the proposed online prediction system.

C. Planning a Manipulator Trajectory

The most popular problem in the robot motion planning is collision avoidance in static and dynamic environments. Starting with potential field approach [25] and elastic band approach [26], many methods targeting on collision-free path planning has been proposed so far. Especially in the manipulation motion planning, random-sampling-based approaches such as probabilistic road map (PRM) and rapidly exploring random trees (RRT) are popular. These approaches are often combined with a concept of configuration space (C-Space) [29], it can calculate a collision-free path for the manipulator in high-dimensional space in real time. In order to solve problems such as further improvement in the algorithm efficiency and guarantee of optimality, RRT-Connect [30] and RRT* [31] have been proposed. However, these algorithms often do not consider temporal aspects, and it is necessary to convert the planned path into the trajectory for applying it to robot control.

In recent years, optimization-based methods are beginning to be reviewed with the background of improving a calculator performance and developing high-speed algorithms. One popular method is a covariant Hamiltonian optimization for motion planning (CHOMP) [32]. It optimizes trajectory costs by using preconditioned gradient descent, called Hamiltonian Monte Carlo method. As approaches with the same motivation, stochastic trajectory optimization for motion planning (STOMP) [33], which uses a stochastic scheme for numerical optimization, and incremental trajectory optimization for motion planning (ITOMP) [34], which extended the CHOMP to the real-time replanning, are also proposed. Further development of these optimization-based approaches, Schulman *et al.* proposed an approach using the sequential convex optimization for solving a nonconvex optimization problem [35]. Kim *et al.* proposed the manipulator

motion planning, which satisfies multiple constraints by using particle swarm optimization (PSO) [36]. These optimization-based methods can directly calculate the trajectory of the robot, and realize a guarantee of collision avoidance and smoothing.

We also adopt the optimization-based approach for trajectory planning. In addition to the requirements of collision avoidance and smoothness, the proposed trajectory generator is required to realize the reduction of the waste time. In order to solve this problem, we introduce a concept of the receding horizon, which is used such as in model predictive control. The concept of the receding horizon is widely used, for example, for task-parameterized motion planning [37], for haptic assistance [38], for physical human–robot interaction [39], and for multi-agent motion planning [40]. The different part of the proposed method is to take a temporal requirement into account explicitly for motion planning by setting a terminal state of the robot in the expanded time space. In addition, the proposed method considers the uncertainty of prediction to deal with the trade-off relationship between the temporal requirement and collision avoidance. By planning the trajectory in consideration of the predicted trajectory of the worker, this strategy enables to simultaneously satisfy two requirements: collision avoidance and the reduction of the waste time.

III. Predicting Worker Motion Using the Adaptive Learning Method

In this section, we summarize the prediction system proposed in [2]. The goal is to model the following two pieces of information to plan the robot's trajectory.

1) Where does the worker work (working position model).
2) How does the worker move in the workspace (moving trajectory model).

The working position model is used to estimate a current task and determine the robot's target position. Conversely, the moving trajectory model is used to plan the robot's trajectory to decrease the risk of robot–worker collisions.

In this paper, we select a modeling approach based on the Gaussian mixture distribution, which is used in the aforementioned models and is given by

$$p(\boldsymbol{x}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m,\ \boldsymbol{\Sigma}_m) \tag{1}$$

where $\boldsymbol{x}$ is the random variable, and M is the mixture number. $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m,\ \boldsymbol{\Sigma}_m)$ is the mth multivariate normal distribution with the mean vector $\boldsymbol{\mu}_m$ and the covariance matrix $\boldsymbol{\Sigma}_m$, and π_m is the mixture coefficient that must satisfy $0 \leq \pi_m \leq 1$ and $\sum_{m=1}^{M} \pi_m = 1$.

A. Estimating the Working Position

The working position (i.e., where the worker works) is stochastically modeled by the Gaussian mixture model (GMM). A stochastic model can estimate the current task robustly against calibration errors while dealing with individual differences.

We now apply Hotelling's T-square method to estimate the current task. When the observed point $\boldsymbol{x}$ is input into a normal distribution $\mathcal{N} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the Mahalanobis distance $D_{\mathrm{M}}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is calculated as

$$D_{\mathrm{M}}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sqrt{(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}. \tag{2}$$

It is known that the square of Mahalanobis distance $D_{\mathrm{M}}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})^2$ follows χ^2 distribution. Additionally, considering the worker velocity, the condition for the worker to start working at working position i is

$$\begin{cases} D_{\mathrm{M}}^2(\boldsymbol{x}_{\mathrm{worker}}^{(t)}, \boldsymbol{\mu}_{\mathrm{task},i}, \boldsymbol{\Sigma}_{\mathrm{task},i}) < a_{\mathrm{th},n}(\alpha) \\ \left|\left| \frac{\boldsymbol{x}_{\mathrm{worker}}^{(t)} - \boldsymbol{x}_{\mathrm{worker}}^{(t-1)}}{\Delta t_{\mathrm{s}}} \right|\right| < v_{\mathrm{th}} \end{cases} \tag{3}$$

where $\boldsymbol{x}_{\mathrm{worker}}^{(t)} \in \mathbb{R}^n$ is the worker position such as a worker's body center and a worker's hand at time t. t is the current time step and Δt_{s} is a measurement period of a sensor for observing the worker position. $\boldsymbol{\mu}_{\mathrm{task},i}$ and $\boldsymbol{\Sigma}_{\mathrm{task},i}$ are the mean vector and the covariance matrix of the ith working position model. This approach can distinguish the current task independent of the variance of the model based on a preset threshold $a_{\mathrm{th},m}(\alpha)$ and v_{th}. v_{th} is a velocity threshold of the worker's motion. $a_{\mathrm{th},m}(\alpha)$ is a percentile threshold of χ^2 distribution. α is an upper significance probability and m is a degree of freedom of χ^2 distribution. In (3), m corresponds to the worker position's dimension n. By installing the threshold $a_{\mathrm{th},m}(\alpha)$, the current task can be distinguished independently of the structure of the model.

The procedure of modeling and estimating the working position is summarized as follows.

1) Using the total number of tasks as the mixture number M_{gmm} and the mean position vectors as the scheduled working positions, an initial GMM is generated. Here, any matrix can be used as the covariance matrix for the initial value.
2) Using the current worker position $\boldsymbol{x}_{\mathrm{worker}}^{(t)}$ and the constructed GMM, the current task i is estimated. At the same time, the worker positions when the worker's velocity is less than a predefined value v_{th} are extracted and stored as sample data.
3) After all tasks are completed, the GMM is updated by the online EM algorithm using the stored sample data.
4) Repeat steps 2 and 3.

The detail of the online EM algorithm is described in Section III-C.

B. Predicting the Worker Trajectory

Worker trajectory is modeled and predicted using the GMR. When using the worker's velocity or acceleration for modeling, the precise prediction becomes difficult because of the sensor noise and the errors involved with differential calculus. In the proposed algorithm, we choose an autoregressive model that uses a history of the worker position to consider the worker's temporal behavior and improve prediction performance. Let $\boldsymbol{x}_c = \boldsymbol{x}_{\mathrm{worker}}^{(t)} \in \mathbb{R}^n$ be a worker's current position and $\boldsymbol{x}_h = (\boldsymbol{x}_{\mathrm{worker}}^{(t-1)}\ \boldsymbol{x}_{\mathrm{worker}}^{(t-2)}\ \cdots \boldsymbol{x}_{\mathrm{worker}}^{(t-d)})^{\mathrm{T}} \in \mathbb{R}^{n\times(d-1)}$ be the worker's position history. d is the order of the autoregression

model. The joint distribution of $\boldsymbol{x}_c$ and $\boldsymbol{x}_h$ is given by

$$p(\boldsymbol{x}_h, \boldsymbol{x}_c) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}_h, \boldsymbol{x}_c | \boldsymbol{\mu}_m,\ \boldsymbol{\Sigma}_m) \tag{4}$$

$$\boldsymbol{\mu}_m = \begin{bmatrix} \boldsymbol{\mu}_m^{\boldsymbol{x}_h} \\ \boldsymbol{\mu}_m^{\boldsymbol{x}_c} \end{bmatrix} \tag{5}$$

$$\boldsymbol{\Sigma}_m = \begin{bmatrix} \boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_h} & \boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_c} \\ \boldsymbol{\Sigma}_m^{\boldsymbol{x}_c \boldsymbol{x}_h} & \boldsymbol{\Sigma}_m^{\boldsymbol{x}_c \boldsymbol{x}_c} \end{bmatrix}. \tag{6}$$

Using this joint distribution, the expectation vector and the variance–covariance matrix of the worker's current position $\boldsymbol{x}_c$, where the worker's position history $\boldsymbol{x}_h$ is given are calculated as

$$E[\boldsymbol{x}_c | \boldsymbol{x}_h] = \sum_{m=1}^{M} h^m(\boldsymbol{x}_h) \boldsymbol{\mu}' \tag{7}$$

$$V[\boldsymbol{x}_c | \boldsymbol{x}_h] = \sum_{m=1}^{M} h^m(\boldsymbol{x}_h) \boldsymbol{\Sigma}' + \sum_{m=1}^{M} h^m(\boldsymbol{x}_h)(\boldsymbol{\mu}' - E[\boldsymbol{x}_c | \boldsymbol{x}_h])(\boldsymbol{\mu}' - E[\boldsymbol{x}_c | \boldsymbol{x}_h])^{\mathrm{T}} \tag{8}$$

where

$$h^m(\boldsymbol{x}_h) = \frac{\pi_m \mathcal{N}(\boldsymbol{x}_h | \boldsymbol{\mu}_m^{\boldsymbol{x}_h},\ \boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_h})}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_h | \boldsymbol{\mu}_k^{\boldsymbol{x}_h},\ \boldsymbol{\Sigma}_k^{\boldsymbol{x}_h \boldsymbol{x}_h})} \tag{9}$$

$$\boldsymbol{\mu}' = \boldsymbol{\mu}_m^{\boldsymbol{x}_c} + \boldsymbol{\Sigma}_m^{\boldsymbol{x}_c \boldsymbol{x}_h} (\boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_h})^{-1} (\boldsymbol{x}_h - \boldsymbol{\mu}_m^{\boldsymbol{x}_h}) \tag{10}$$

$$\boldsymbol{\Sigma}' = \boldsymbol{\Sigma}_m^{\boldsymbol{x}_c \boldsymbol{x}_c} - \boldsymbol{\Sigma}_m^{\boldsymbol{x}_c \boldsymbol{x}_h} (\boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_h})^{-1} \boldsymbol{\Sigma}_m^{\boldsymbol{x}_h \boldsymbol{x}_c}. \tag{11}$$

When making the prediction, the current position $\boldsymbol{x}_c^{(t+1)}$ at step $t+1$ is calculated using

$$\boldsymbol{x}_c^{(t+1)} = E[\boldsymbol{x}_c^{(t+1)} | \boldsymbol{x}_h^{(t+1)}]$$

$$\boldsymbol{x}_h^{(t+1)} = \left(\boldsymbol{x}_c^{(t)}\ \boldsymbol{x}_c^{(t-1)}\ \cdots \boldsymbol{x}_c^{(t+1-d)} \right)^{\mathrm{T}}. \tag{12}$$

The calculation using (12) is repeated until the length of the predicted trajectory reaches max prediction length T_p. The trajectory prediction procedure is summarized in Algorithm 1. The worker's predicted trajectory is expressed as a set of Gaussian distributions. For example, the worker's predicted position at step t becomes

$$\mathcal{N}^{(t)} = \mathcal{N}(\boldsymbol{\mu}_{\mathrm{worker}}^{(t)},\ \boldsymbol{\Sigma}_{\mathrm{worker}}^{(t)}) \tag{13}$$

where $\boldsymbol{\mu}_{\mathrm{worker}}^{(t)}$ is the mean vector of the worker's predicted position and $\boldsymbol{\Sigma}_{\mathrm{worker}}^{(t)}$ is the covariance matrix of the worker's predicted position at step t.

C. Online Learning Algorithm

An online learning algorithm for the Gaussian mixture distribution is installed to update the model parameters. The EM algorithm, which is a common way to learn Gaussian parameters, is extended to an incremental version and the parameters are updated sequentially. The procedure is as follows.

Algorithm 1: Online Prediction of Worker's Motion Trajectory.

t is the current time, $\boldsymbol{x}_c^{(t)}$ is the current position
$\boldsymbol{x}_h^{(t)}$ is the position history.
while $k = 1$ *to* T_p **do**
$\boldsymbol{x}_h^{(t+k)} = \left(\boldsymbol{x}_c^{(t+k-1)}\ \boldsymbol{x}_c^{(t+k-2)}\ \cdots \boldsymbol{x}_c^{(t+k-d-1)} \right)^{\mathrm{T}}$
$\boldsymbol{\mu}_{\mathrm{worker}}^{(t+k)} = E[\boldsymbol{x}_c^{(t+k)} | \boldsymbol{x}_h^{(t+k)}]$
$\boldsymbol{\Sigma}_{\mathrm{worker}}^{(t+k)} = V[\boldsymbol{x}_c^{(t+k)} | \boldsymbol{x}_h^{(t+k)}]$
$\boldsymbol{x}_c^{(t+k)} = E[\boldsymbol{x}_c^{(t+k)} | \boldsymbol{x}_h^{(t+k)}]$
end while
Worker's motion trajectory is
$\left(\mathcal{N}^{(t)}, \mathcal{N}^{(t+1)}, \cdots \mathcal{N}^{(t+T_p)} \right)$

1) (E-step) Using the parameters at step k, the burden rate γ_{lm} is calculated as

$$\gamma_{lm} = \frac{\mathcal{N}(\boldsymbol{x}_l | \boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})}{\sum_{m=1}^{M} \pi_m^{(k)} \mathcal{N}(\boldsymbol{x}_l | \boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})} \pi_m^{(k)}. \tag{14}$$

2) (M-step) Each parameter is estimated from the burden rate γ_{lm} as

$$\pi_{\mathrm{est},m} = \frac{N_s}{S_{\gamma,m}} \tag{15}$$

$$\boldsymbol{\mu}_{\mathrm{est},m} = \frac{1}{S_{\gamma,m}} \sum_{l=1}^{N_s} \gamma_{lm} \boldsymbol{x}_l \tag{16}$$

$$\boldsymbol{\Sigma}_{\mathrm{est},m} = \frac{1}{S_{\gamma,m}} \sum_{l=1}^{N_s} \gamma_{lm} (\boldsymbol{x}_l - \boldsymbol{\mu}_m^{(k)})(\boldsymbol{x}_l - \boldsymbol{\mu}_m^{(k)})^{\mathrm{T}} \tag{17}$$

where N_s is the total number of sample data and $S_{\gamma,m}$ is given by

$$S_{\gamma,m} = \sum_{l=1}^{N_s} \gamma_{lm}. \tag{18}$$

3) Update the parameters using the forgetting factor $\eta(k)$ as

$$\pi_m^{(k+1)} = (1 - \eta(k)) \pi_m^{(k)} + \eta(k) \pi_{\mathrm{est},m} \tag{19}$$

$$\boldsymbol{\mu}_m^{(k+1)} = (1 - \eta(k)) \boldsymbol{\mu}_m^{(k)} + \eta(k) \boldsymbol{\mu}_{\mathrm{est},m} \tag{20}$$

$$\boldsymbol{\Sigma}_m^{(k+1)} = (1 - \eta(k)) \boldsymbol{\Sigma}_m^{(k)} + \eta(k) \boldsymbol{\Sigma}_{\mathrm{est},m}. \tag{21}$$

We design the $\eta(k)$ as

$$\eta(k) = (k+2)^{-\beta} \tag{22}$$

where β is the constant scaler value, which means the degree of update against past parameters.

In the incremental learning algorithm, it is often assumed that the mixture number is constant. However, selecting a proper mixture number for the observed sample is important for precise modeling. In the proposed system, we propose a method that adds or deletes a cluster of mixed normal distributions based on the sample data as follows.

1) Unit Addition: When the new sample data $\boldsymbol{x}_l$ are given, we calculate the $p_{\mathrm{add}}(\boldsymbol{x}_l)$ as

$$p_{\mathrm{add}}(\boldsymbol{x}_l) = \min_{1\le m\le M} D^2_{\mathrm{M}}(\boldsymbol{x}_l,\ \boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)}) \tag{23}$$

where $p_{\mathrm{add}}(\boldsymbol{x}_l)$ means how far the sample data $\boldsymbol{x}_l$ is from the current model. Using the percentile threshold $a_{\mathrm{th},m}(\alpha)$, a new normal distribution is generated when the condition $p(\boldsymbol{x}_l) > a_{\mathrm{th},m}(\alpha)$ is satisfied. The new mixture coefficient and covariance matrix are set as the predefined value $\pi_{\mathrm{ini}},\ \boldsymbol{\Sigma}_{\mathrm{ini}}$.

2) Unit Deletion: For each cluster m that constitutes the current model, we calculate $p_{\mathrm{del}}(m)$ as

$$p_{\mathrm{del}}(m) = \min_{1\le l\le N_s} D^2_{\mathrm{M}}(\boldsymbol{x}_l,\ \boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)}) \tag{24}$$

where $p_{\mathrm{del}}(m)$ denotes how much the cluster m contributes to express the observed samples. Using the percentile threshold $a_{\mathrm{th},m}(\alpha)$, the cluster m is deleted when the condition $p(\boldsymbol{x}_l) > a_{\mathrm{th},m}(\alpha)$ is satisfied.

In the unit addition and deletion algorithm, m corresponds to the dimension of the worker position history $n \times d$. By installing the threshold $a_{\mathrm{th},m}(\alpha)$, it is possible to determine the threshold not dependent on the sample data representing the model. The proposed online learning algorithm is summarized in Algorithm 2.

Algorithm 2: Incremental Update of Gaussian parameters.

k is the current step
while $l = 1$ *to* N_s **do**
  $p_{\mathrm{add}}(\boldsymbol{x}_l) \Leftarrow \min_{1\le m\le M}\ D^2_{\mathrm{M}}(\boldsymbol{x}_l|\boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})$
  if $p_{\mathrm{add}}(\boldsymbol{x}_l) > a_{\mathrm{th},n\times d}(\alpha)$ **then**
    $M \Leftarrow M + 1$
    $\pi_{M+1} \Leftarrow \pi_{\mathrm{ini}}$
    $\boldsymbol{\mu}_{M+1} \Leftarrow \boldsymbol{x}_l$
    $\boldsymbol{\Sigma}_{M+1} \Leftarrow \boldsymbol{\Sigma}_{\mathrm{ini}}$
  end if
end while
Normalize all $\pi_m^{(k)}$
while $m = 1$ *to* M **do**
  while $l = 1$ *to* N_s **do**
    $\gamma_{lm} \Leftarrow \frac{\mathcal{N}(\boldsymbol{x}_l|\boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})}{\sum_{m=1}^{M}\pi_m^{(k)}\mathcal{N}(\boldsymbol{x}_l|\boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})}\pi_m^{(k)}$
  end while
  while $l = 1$ *to* N_s **do**
    $S_{\gamma,m} \Leftarrow S_{\gamma,m} + \gamma_{lm}$
    $S_{\mu} \Leftarrow S_{\mu} + \gamma_{lm}\boldsymbol{x}_l$
    $S_{\Sigma} \Leftarrow S_{\Sigma} + \gamma_{lm}(\boldsymbol{x}_l - \boldsymbol{\mu}_m^{(k)})(\boldsymbol{x}_l - \boldsymbol{\mu}_m^{(k)})^{\mathrm{T}}$
    $p_{\mathrm{del}}(m) \Leftarrow \min_{1\le l\le N_s}\ D^2_{\mathrm{M}}(\boldsymbol{x}_l|\boldsymbol{\mu}_m^{(k)},\ \boldsymbol{\Sigma}_m^{(k)})$
  end while
  if $p_{\mathrm{del}}(m) > a_{\mathrm{th},n\times d}(\alpha)$ **then**
    Delete the mth cluster
  $M \Leftarrow M - 1$
  else
    $\pi_{\mathrm{est},m} \Leftarrow N_s/S_{\gamma,m}$
    $\boldsymbol{\mu}_{\mathrm{est},m} \Leftarrow S_{\mu}/S_{\gamma,m}$
    $\boldsymbol{\Sigma}_{\mathrm{est},m} \Leftarrow S_{\Sigma}/S_{\gamma,m}$
  end if
end while
Normalize all $\pi_{\mathrm{est},m}$
Calculate $\pi_m^{(k+1)}\ \boldsymbol{\mu}_m^{(k+1)}\ \boldsymbol{\Sigma}_m^{(k+1)}$ by (19), (20), (21)

IV. Online Trajectory Planning

Here, we consider the robot's trajectory planning using the predicted motion of the worker given by the worker motion predictor. There are two advantages of using probabilistic long-term predicted worker's motion for the trajectory planning, which are as follows.

1) Temporal requirements can be considered explicitly for the trajectory planning. This strategy reduces the waste time for workers to wait for the robot movement.
2) An irregularity of the worker's motion can be considered for the trajectory planning.

Considering the covariance of predicted worker's positions, the robot enables to avoid collision with the worker even when the worker moves in an unexpected way. In the following, we propose an online trajectory planning method that effectively utilizes the predicted worker's motion.

A. Online Trajectory Planning Based on a Receding Horizon Scheme

In this paper, We define the trajectory that simultaneously satisfies the following three requirements as an optimal trajectory.

1) The endpoint of the manipulator arrives at the scheduled target position at the moment when the worker needs support by the robot.
2) The manipulator avoids colliding with the worker by considering the probability distribution of the predicted worker position at each time step.
3) The manipulator moves under the preset velocity and acceleration limitations.

Condition 1) is aimed at decreasing the time wasted by delaying the robot's tasks, which improves the work-time efficiency. Condition 2) is aimed at decreasing the risk of collision by avoiding areas where the possibility of collision is high. Condition 3) is aimed at smoothing the calculated robot's trajectory to enable the robot to certainly follow the desired trajectory.

For the online trajectory generator to satisfy the aforementioned three conditions, we solve an optimization problem based on the concept of the receding horizon. By installing the receding horizon concept, the trajectory generator can effectively manage the temporal requirements. Fig. 2 illustrates the concept of the proposed trajectory planning method. Considering that the predicted trajectory of the worker is given, the proposed method plans the trajectory of the robot in the space extended along the time axis. Because the probability distributions of the predicted position of the worker are given at each time step, the state of the robot with low possibility of collision can be determined at each time step. Furthermore, setting the target position in the space with the time axis allows us to calculate the robot trajectory that reaches the target position at the assumed target time. The predicted trajectory of the worker is updated at each measurement period of the sensor, so the robot's optimal trajectory is updated at each sensor measurement.

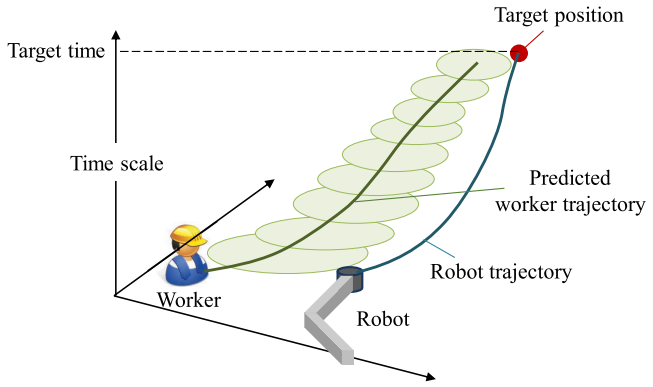


Fig. 2. Concept of the trajectory planning with the receding horizon scheme considering probabilistic distributions of predicted worker positions.

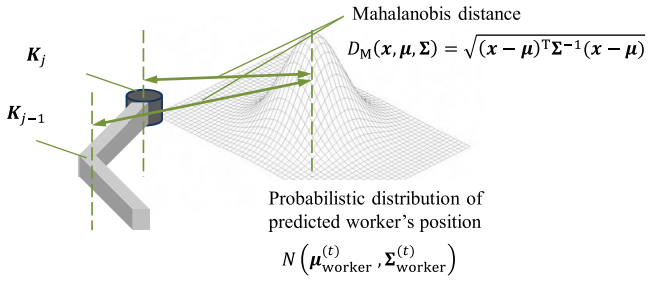


Fig. 3. Artificial potential field built from a probabilistic distribution of the worker's predicted position.

B. Cost Functions for Exploiting Worker's Predicted Trajectory

Here, we define the cost function J by following one standard formulation of the receding horizon scheme as

$$J = \varphi(\boldsymbol{q}(t+T_o)) + \int_t^{t+T_o} \left(L_1(\dot{\boldsymbol{q}}(k)) + L_2(\boldsymbol{q}(k))\right) dk \quad (25)$$

where T_o is a length of optimized trajectory, $\boldsymbol{q} = (\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is a state vector of the manipulator, $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_{N_j})$ is a vector composed of joint angles of the manipulator, and N_j is the degrees of freedom of the manipulator. $\varphi(\boldsymbol{q}(t+T_o))$ is a cost function that specifies the state of the robot at the end of the trajectory. It becomes a constraint of the terminal state and relates to the Condition 1). $L_1(\dot{\boldsymbol{q}}(k))$ and $L_2(\boldsymbol{q}(k))$ are cost functions that specify each state constituting the trajectory of the robot. It performs as functions of the collision avoidance and smoothing and relates to the Conditions 2) and 3).

In the proposed method, same as the predicted trajectory of the worker, the robot's trajectory is discretized with the sensor measurement period. The discretized cost function J is constituted as

$$J = \varphi(\boldsymbol{q}^{(t+T_o)}) + \Sigma_{k=t}^{t+T_o} \left(L_1(\boldsymbol{q}^{(k)}, \boldsymbol{q}^{(k-1)}) + L_2(\boldsymbol{q}^{(k)})\right) \quad (26)$$

where the functions composing cost function J are

$$\varphi(\boldsymbol{q}^{(k)}) = \frac{1}{2}\left(\boldsymbol{K}_{N_j}(\boldsymbol{q}^{(k)}) - \boldsymbol{x}_{\mathrm{target}}\right)^{\mathrm{T}} \times R\left(\boldsymbol{K}_{N_j}(\boldsymbol{q}^{(k)}) - \boldsymbol{x}_{\mathrm{target}}\right) \quad (27)$$

$$L_1(\boldsymbol{q}^{(k)}, \boldsymbol{q}^{(k-1)}) = \frac{1}{2}\sum_{j=1}^{N_j} r_j (B_{\mathrm{vel},j}(\dot{\theta}_j^{(k)}) + B_{\mathrm{acc},j}(\ddot{\theta}_j^{(k)})) \quad (28)$$

$$L_2(\boldsymbol{q}^{(k)}) = w \sum_{j=1}^{N_j} \frac{1}{D_{\mathrm{M}}\left(\boldsymbol{K}_j(\boldsymbol{\theta}^{(k)}), \boldsymbol{\mu}^{(k)}_{\mathrm{worker}}, \boldsymbol{\Sigma}^{(k)}_{\mathrm{worker}}\right)}. \quad (29)$$

The term $\varphi(\boldsymbol{q}^{(k)})$ ensures convergence to the target state at the time step $t+T_o$. $\boldsymbol{K}_j(\boldsymbol{q}^{(k)})$ is the position and velocity of the jth joint at step k and is calculated using a robot's kinematics. In particular, $\boldsymbol{K}_{N_j}(\boldsymbol{q}^{(k)})$ corresponds to the endpoint state. $\boldsymbol{x}_{\mathrm{target}}$ is the target position and velocity of the robot's endpoint, and R is the diagonal matrix whose diagonal components are weighting factors for the components of the state vector. In this case, zero is set as the target velocity.

$L_1(\boldsymbol{q}^{(k)}, \boldsymbol{q}^{(k-1)})$ is a penalty function for considering the inequality constraints of velocity and acceleration limits. $B_{\mathrm{vel},j}(\dot{\theta}_j^{(k)})$ and $B_{\mathrm{acc},j}(\ddot{\theta}_j^{(k)})$ become

$$B_{\mathrm{vel},j}(\dot{\theta}_j) = \begin{cases} 0, & (||\dot{\theta}_j|| \le \dot{\theta}_{\max,j}) \\ \left(||\dot{\theta}_j|| - \dot{\theta}_{\max,j}\right)^2, & (||\dot{\theta}_j|| > \dot{\theta}_{\max,j}) \end{cases} \quad (30)$$

$$B_{\mathrm{acc},j}(\ddot{\theta}_j) = \begin{cases} 0, & (||\ddot{\theta}_j|| \le \ddot{\theta}_{\max,j}) \\ \left(||\ddot{\theta}_j|| - \ddot{\theta}_{\max,j}\right)^2, & (||\ddot{\theta}_j|| > \ddot{\theta}_{\max,j}) \end{cases}. \quad (31)$$

In the proposed system, we set the maximum angular velocities $\dot{\theta}_{\max,j}$ and the maximum angular accelerations $\ddot{\theta}_{\max,j}$ for each joint. r_j are the weighting factors for the constraints of each joint.

The term $L_2(\boldsymbol{q}^{(k)})$ serves to avoid a collision with the worker based on the distribution of the predicted worker position. w is a weighting coefficient of this cost function. Using the Mahalanobis distance between the distributions of the predicted worker position $\mathcal{N}(\boldsymbol{\mu}^{(k)}_{\mathrm{worker}}, \boldsymbol{\Sigma}^{(k)}_{\mathrm{worker}})$ at step k and each joint position $\boldsymbol{K}_j(\boldsymbol{\theta}^{(k)})$, an artificial potential field is constituted as shown in Fig. 3. Using the Mahalanobis distance, the artificial potential is wider in the direction of larger variance in the predicted position. Therefore, in this direction of high uncertainty in the worker position, it is possible to widely avoid the worker and decrease the risk of the collision.

Note that the collision avoidance is imposed soft constraint to allow the robot to approach the worker for support. Since the collaborative robot often has physical interactions with workers (e.g., handover and hand guiding), it is impossible for the robot to achieve the intended task simply by taking a distance from the worker. In this formulation, there is a tradeoff relationship between the term $\varphi(\boldsymbol{q}^{(k)})$ and the term $L_2(\boldsymbol{q}^{(k)})$, and the priority of these terms depends on the ratio of coefficients R and w. As another point to note, only the collision with the joint position is considered in the cost function $L_2(\boldsymbol{q}^{(k)})$ in this formulation. However, collision with the link can also be considered by setting some reference points not only on each joint but on each link.

C. Solving the Optimization Problem

The optimization problem to be solved in this system is formulated as follows:

$$\begin{aligned} &\text{minimize} && J \\ &\text{subject to} && \dot{\boldsymbol{q}} = f(\boldsymbol{q}, \boldsymbol{u}) \\ & && \boldsymbol{q}(t) = \boldsymbol{q}_{\mathrm{cur}} \end{aligned}$$

where f is a nonlinear dynamics of the robot, $\boldsymbol{u}$ is a input vector for the robot, and $\boldsymbol{q}(t)$ is the initial state of the trajectory and corresponds to the robot's current state $\boldsymbol{q}_{\mathrm{cur}}$. This formulation means that the calculated trajectory of the robot is limited by its dynamics and initial state. By solving this optimization problem with two equality constraints, we calculate the optimal state of the robot at each time, that is, the optimal trajectory of the robot.

To solve the optimization problem with the equality constraints described previously, we use the Lagrange multiplier method same as [41]. By installing the Lagrange vector $\boldsymbol{\lambda}$ whose dimension corresponds to the state vector $\boldsymbol{q}$, the cost function (25) with equality constraints becomes

$$\begin{aligned} J &= \varphi(\boldsymbol{q}(t+T_o)) \\ &\quad + \int_t^{t+T_o} \big(L_1(\dot{\boldsymbol{q}}(k)) + L_2(\boldsymbol{q}(k)) + \boldsymbol{\lambda}^{\mathrm{T}}(k)(f(\boldsymbol{q}(k), \\ &\quad \times u(k)) - \dot{\boldsymbol{q}}(k))\big)\, dk \\ &= \varphi(\boldsymbol{q}(t+T_o)) + \int_t^{t+T_o} (H(\boldsymbol{q}(k), \boldsymbol{u}(k), \boldsymbol{\lambda}(k)) \\ &\quad - \boldsymbol{\lambda}^{\mathrm{T}}(k)\dot{\boldsymbol{q}}(k))\big)\, dk \end{aligned} \tag{32}$$

where H is the Hamiltonian and is defined as

$$H(\boldsymbol{q}, \boldsymbol{u}, \boldsymbol{\lambda}) = L_1(\dot{\boldsymbol{q}}) + L_2(\boldsymbol{q}) + \boldsymbol{\lambda}^{\mathrm{T}} f(\boldsymbol{q}, \boldsymbol{u}). \tag{33}$$

Here, total derivative of J becomes

$$\begin{aligned} \delta J &= \left(\frac{\partial \varphi}{\partial \boldsymbol{q}}(\boldsymbol{q}(t+T_o)) - \boldsymbol{\lambda}^{\mathrm{T}}(t+T_o)\right)\delta \boldsymbol{q} \\ &\quad + \int_t^{t+T_o} \left(\left(\frac{\partial H}{\partial \boldsymbol{q}}(\boldsymbol{q}(k), \boldsymbol{u}(k), \boldsymbol{\lambda}(k)) + \boldsymbol{\lambda}^{\mathrm{T}}(k)\right)\delta \boldsymbol{q} \right. \\ &\quad \left. + \frac{\partial H}{\partial \boldsymbol{u}}(\boldsymbol{q}(k), \boldsymbol{u}(k), \boldsymbol{\lambda}(k))\delta \boldsymbol{u}\right) dk \end{aligned} \tag{34}$$

where assuming that $\boldsymbol{\lambda}^{\mathrm{T}}(t+T_o) = \frac{\partial \varphi}{\partial \boldsymbol{q}}(\boldsymbol{q}(t+T_o))$ and $\boldsymbol{\lambda}^{\mathrm{T}}(k) = -\frac{\partial H}{\partial \boldsymbol{q}}(\boldsymbol{q}(k), \boldsymbol{u}(k), \boldsymbol{\lambda}(k))$ are satisfied, (34) becomes

$$\delta J = \int_t^{t+T_o} \frac{\partial H}{\partial \boldsymbol{u}}(\boldsymbol{q}(k), \boldsymbol{u}(k), \boldsymbol{\lambda}(k))\delta \boldsymbol{u} dk. \tag{35}$$

This means that $\frac{\partial H}{\partial \boldsymbol{u}}$ corresponds to the gradient of the cost function J.

Discretizing the optimization problem conforming with (26), equations that the optimal solution should satisfy are summarized as

$$\boldsymbol{q}^{(k+1)} = \boldsymbol{q}^{(k)} + f(\boldsymbol{q}^{(k)}, \boldsymbol{u}^{(k)})\Delta t_{\mathrm{s}} \tag{36}$$

$$\boldsymbol{q}^{(t)} = \boldsymbol{q}_{\mathrm{cur}} \tag{37}$$

$$\boldsymbol{\lambda}^{(k)} = \boldsymbol{\lambda}^{(k+1)} - \left(\frac{\partial H}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(k+1)}, \boldsymbol{u}^{(k)}, \boldsymbol{\lambda}^{(k+1)}) \tag{38}$$

$$\boldsymbol{\lambda}^{(t+T_o)} = \left(\frac{\partial \varphi}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+T_o)}). \tag{39}$$

The procedure for calculating the online trajectory is shown in Algorithm 3. For input, predicted worker's trajectory $\left(\mathcal{N}^{(t)}, \mathcal{N}^{(t+1)}, \ldots \mathcal{N}^{(t+T_p)}\right)$, the target position $\boldsymbol{x}_{\mathrm{target}}$, and the robot's current state $\boldsymbol{q}_{\mathrm{cur}}$ are given. First, the set of input vectors $\boldsymbol{u} = (\boldsymbol{u}^{(t)}, \boldsymbol{u}^{(t+1)}, \ldots, \boldsymbol{u}^{(t+T_o-1)})$ are initialized and $\boldsymbol{q}_{\mathrm{cur}}$ is set to $\boldsymbol{q}^{(t)}$. Note that the optimization length T_o is smaller than the prediction length T_p. Given the initial state and the set of control input vectors, the set of state vectors $\boldsymbol{q} = (\boldsymbol{q}^{(t)}, \boldsymbol{q}^{(t+1)}, \ldots, \boldsymbol{q}^{(t+T_o)})$ is calculated using (36). Next, using $\boldsymbol{\lambda}^{(t+T_o)}$ calculated from (39), the set of Lagrangian vectors $\boldsymbol{\lambda} = (\boldsymbol{\lambda}^{(t)}, \boldsymbol{\lambda}^{(t+1)}, \ldots, \boldsymbol{\lambda}^{(t+T_o)})$ is calculated in the reverse time direction. Using these results, the gradient $\frac{\partial H}{\partial \boldsymbol{u}}$ is calculated and the set of input vectors are updated as

$$\boldsymbol{u} \Leftarrow \boldsymbol{u} + c\boldsymbol{s} \tag{40}$$

where c is a step size of the gradient descent. $\boldsymbol{s}$ is a step direction and becomes

$$\boldsymbol{s} = \begin{pmatrix} \left(\frac{\partial H}{\partial \boldsymbol{u}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+1)}, \boldsymbol{u}^{(t)}, \boldsymbol{\lambda}^{(t+1)}) \\ \left(\frac{\partial H}{\partial \boldsymbol{u}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+2)}, \boldsymbol{u}^{(t+1)}, \boldsymbol{\lambda}^{(t+2)}) \\ \vdots \\ \left(\frac{\partial H}{\partial \boldsymbol{u}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+T_o)}, \boldsymbol{u}^{(t+T_o-1)}, \boldsymbol{\lambda}^{(t+T_o)}) \end{pmatrix}. \tag{41}$$

This sequential calculation is repeated until the condition $\Sigma_{k=t}^{t+T_o} |\frac{\partial H}{\partial \boldsymbol{u}}(\boldsymbol{q}^{(k+1)}, \boldsymbol{u}^{(k)}, \boldsymbol{\lambda}^{(k+1)})| < \epsilon$ is satisfied. ϵ is a small constant for determining the convergence of the sequence calculation. Using this optimal input set, we obtain the optimal trajectory $\left(\boldsymbol{q}^{(t)}, \boldsymbol{q}^{(t+1)}, \ldots, \boldsymbol{q}^{(t+T_o)}\right)$.

V. Applying the Proposed System to an Actual Assembly Scenario

The proposed system can inherently cover various assembly processes. The worker motion predictor can be applied to both two-dimensional and three-dimensional cases depending on the definition of the worker position $\boldsymbol{x}_{\mathrm{worker}}^{(t)}$. The online trajectory generator can be applied to various manipulators by changing the definition of the degree of freedom N_j and the robot's dynamics f. In the following, we target an assembly process where the two-link planar manipulator performs a delivery task in the two-dimensional space. The worker motion predictor described in Section III and the online trajectory generator described in Section IV are installed and mentioned in detail assuming the two-dimensional case.

A. Application to a Process of Performing Delivery Work in Two-Dimensional Space

In this paper, we apply the proposed motion planning system to a two-link manipulator, which is used in an actual assembly process. Fig. 4 shows the two-link planar manipulator used for

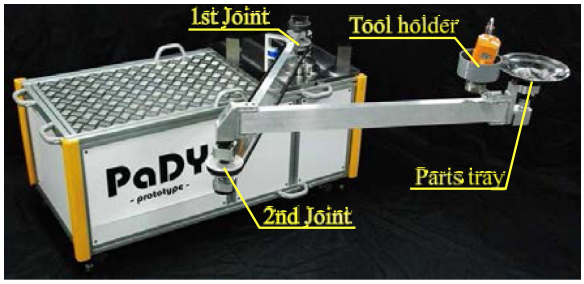


Fig. 4. Two-link planar manipulator [42].

Algorithm 3: Online Trajectory Generator.

Predicted worker's trajectory is $\left(\mathcal{N}^{(t)}, \mathcal{N}^{(t+1)}, \ldots \mathcal{N}^{(t+T_p)}\right)$
The target position is $\boldsymbol{x}_{\mathrm{target}}$
The current state of the robot is $\boldsymbol{q}_{\mathrm{cur}}$
Initialize the set of input vectors $\boldsymbol{u}$
$\boldsymbol{q}^{(t)} \Leftarrow \boldsymbol{q}_{\mathrm{cur}}$
while $\Sigma_{k=t}^{t+T_o} |\frac{\partial H}{\partial \boldsymbol{u}}(\boldsymbol{q}^{(k+1)}, \boldsymbol{u}^{(k)}, \boldsymbol{\lambda}^{(k+1)})| < \epsilon$ **do**
  while $k = 1$ *to* T_o **do**
    $\boldsymbol{q}^{(t+k)} \Leftarrow \boldsymbol{q}^{(t+k-1)} + f(\boldsymbol{q}^{(t+k-1)}, \boldsymbol{u}^{(t+k-1)})\Delta t_{\mathrm{s}}$
  end while
  while $k = T_o$ *to* 1 **do**
    $\boldsymbol{\lambda}^{(t+k-1)} \Leftarrow \boldsymbol{\lambda}^{(t+k)} - \left(\frac{\partial H}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+k)}, \boldsymbol{u}^{(t+k)}, \boldsymbol{\lambda}^{(t+k+1)})$
  end while
  while $i = 1$ *to* T_o **do**
    $s_i \Leftarrow \left(\frac{\partial H}{\partial \boldsymbol{u}}\right)^{\mathrm{T}} (\boldsymbol{q}^{(t+i-1)}, \boldsymbol{u}^{(t+i-1)}, \boldsymbol{\lambda}^{(t+i)})$
  end while
  $\boldsymbol{u} \Leftarrow \boldsymbol{u} + c\boldsymbol{s}$
end while
while $k = 1$ *to* T_o **do**
  $\boldsymbol{q}^{(t+k)} \Leftarrow \boldsymbol{q}^{(t+k-1)} + f(\boldsymbol{q}^{(t+k-1)}, \boldsymbol{u}^{(t+k-1)})\Delta t_{\mathrm{s}}$
end while
Optimal trajectory is $\left(\boldsymbol{q}^{(t)}, \boldsymbol{q}^{(t+1)}, \ldots \boldsymbol{q}^{(t+T_o)}\right)$

this system. This robot has two joints and a parts tray attached to the robot's end effector. The details of its hardware design are given in [42]. In the assembly process, the robot delivers the parts and tools to the worker. The details of delivery tasks are described in the next section.

Fig. 5 shows the worker motion predictor applied in the two-dimensional space. In this case, the worker position $\boldsymbol{x}_{\mathrm{worker}}^{(t)} \in \mathbb{R}^2$ is a body center of the worker. Therefore, the distribution of ith working position $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{task},i},\ \boldsymbol{\Sigma}_{\mathrm{task},i})$ and the distribution of the predicted worker' position $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{worker}}^{(t)},\ \boldsymbol{\Sigma}_{\mathrm{worker}}^{(t)})$ are also two-dimensional Gaussian distribution.

We define the target time of the robot as the predicted arrival time of the worker T_a at ith working position. The predicted arrival time T_a is the instant when the latest predicted worker position satisfies the condition of the next working position model at the first time. Using (3), the predicted arrival time T_a is

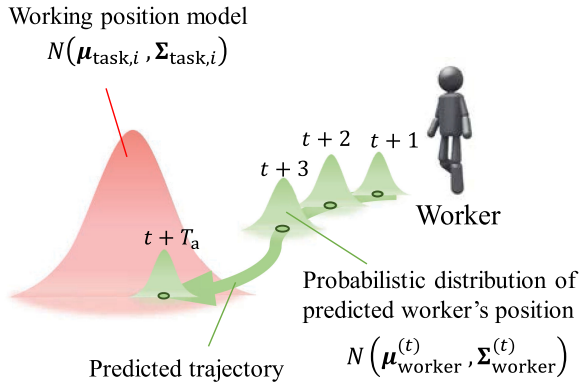


Fig. 5. Worker motion predictor in the two-dimensional case. The red distribution is the probabilistic distribution of the next working position. The green arrow denotes the predicted trajectory and is constructed based on distributions of the worker's position predicted for each time step.

determined from the following condition:

$$\begin{cases} D_{\mathrm{M}}^2(\boldsymbol{\mu}_{\mathrm{worker}}^{(t+T_a)}, \boldsymbol{\mu}_{\mathrm{task},i}, \boldsymbol{\Sigma}_{\mathrm{task},i}) < a_{\mathrm{M}} \\ \left|\left| \frac{\boldsymbol{\mu}_{\mathrm{worker}}^{(t+T_a)} - \boldsymbol{\mu}_{\mathrm{worker}}^{(t+T_a-1)}}{\Delta t_{\mathrm{s}}} \right|\right| < v_{\mathrm{th}} \end{cases}. \tag{42}$$

In this system, the predicted arrival time T_a satisfies $T_a \leq T_p$ and corresponds to the trajectory length T_o. Then, the robot's target position of task i used in (27) is calculated based on the working position model as

$$\boldsymbol{x}_{\mathrm{target},i} = \boldsymbol{\mu}_{\mathrm{task},i} + \boldsymbol{x}_{\mathrm{offset},i} \tag{43}$$

where $\boldsymbol{x}_{\mathrm{offset},i}$ is a predefined constant for the task i based on the process sheet given in advance.

To install the online trajectory generator into the target process, we derive the dynamics of the robot. The dynamical model of the two-link planar manipulator is expressed as

$$M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \boldsymbol{\tau} \tag{44}$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2)$ is the joint angle, and $\boldsymbol{\tau} = (\tau_1, \tau_2)$ is the input torque to each joint of the manipulator. $M(\boldsymbol{\theta})$ is the inertia term and $C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the Coriolis term. Because this robot moves only in the horizontal space, the gravity term can be ignored in this case. Let the state vector be $\boldsymbol{q} = (\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ and the input be $\boldsymbol{u} = \boldsymbol{\tau}$, the discretized state equation then becomes

$$f(\boldsymbol{q}, \boldsymbol{u}) = \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ -M(\boldsymbol{\theta})^{-1} C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + M(\boldsymbol{\theta})^{-1} \boldsymbol{u} \end{bmatrix}. \tag{45}$$

This discretized dynamics f is used for (36).

B. Procedure to Determine the Control Input

Fig. 6 shows the flowchart used to calculate the control input of the robot. First, the current worker position $\boldsymbol{x}_{\mathrm{worker}}^{(t)}$ observed by the sensor is sent to the worker motion predictor. It stores the worker position as history and estimates the current task i using the condition (3) and predicts the worker's trajectory $\left(\mathcal{N}^{(t)}, \mathcal{N}^{(t+1)}, \ldots \mathcal{N}^{(t+T_p)}\right)$ using Algorithm 1. The estimated current task i and the offset position $\boldsymbol{x}_{\mathrm{offset},i}$ are sent to the target position determiner. It decides the target delivery position $\boldsymbol{x}_{\mathrm{offset}}$

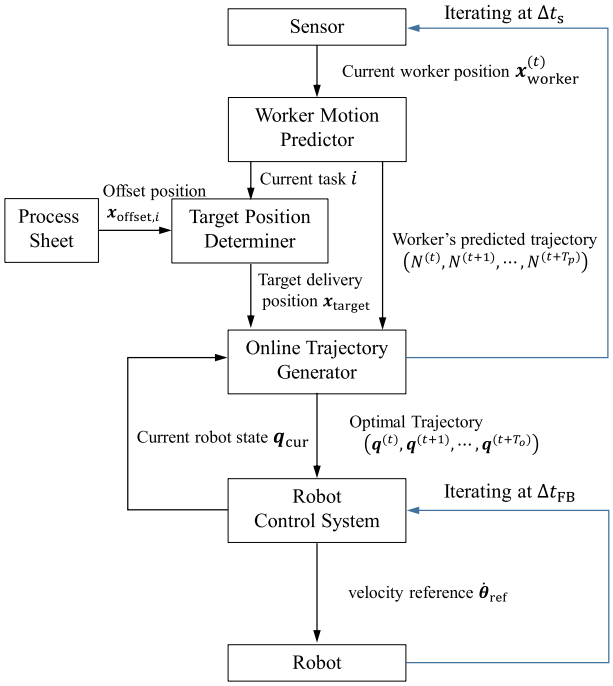


Fig. 6. System flowchart for determining the control input for the robot.

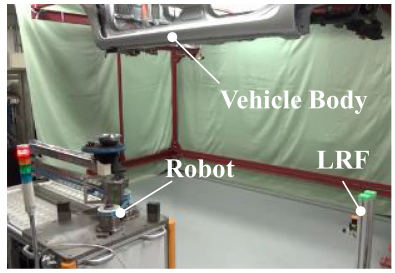


Fig. 7. Experimental environment.

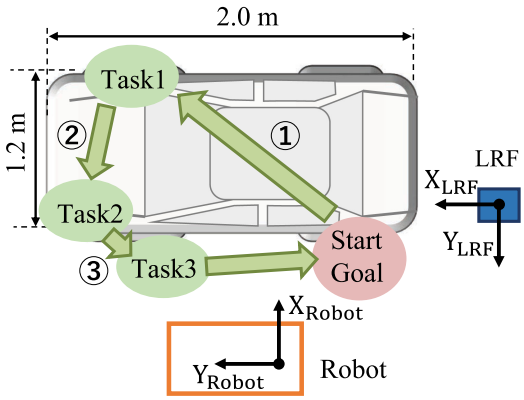


Fig. 8. Top view of the experimental setup.

using (43). Finally, the target delivery position $\boldsymbol{x}_{\mathrm{offset}}$ and the predicted worker's trajectory $\left(\mathcal{N}^{(t)}, \mathcal{N}^{(t+1)}, \ldots \mathcal{N}^{(t+T_p)}\right)$ are sent to the online trajectory generator. Using the input from the worker motion predictor and the current robot state $\boldsymbol{q}_{\mathrm{cur}}$ from the robot control system, the online trajectory generator optimizes the cost function and calculates the optimal trajectory $\left(\boldsymbol{q}^{(t)}, \boldsymbol{q}^{(t+1)}, \ldots \boldsymbol{q}^{(t+T_o)}\right)$ using Algorithm 3. This calculation is repeated at the sensor measurement period Δt_{s} to update the robot's trajectory.

After obtaining the optimal trajectory from the online trajectory generator, the robot control system calculates the velocity reference $\dot{\boldsymbol{\theta}}_{\mathrm{ref}}$ for the robot. Because the sensor measurement period Δt_{s} often becomes larger than the feedback period Δt_{FB}, the robot control system must determine the velocity reference for each feedback cycle based on the calculated optimal trajectory. Using linear interpolation, the velocity reference $\dot{\boldsymbol{\theta}}_{\mathrm{ref}}$ to follow the calculated optimal trajectory is determined as follows:

$$\boldsymbol{q}_{\mathrm{des}} = \boldsymbol{q}^{(t)} + \left(\boldsymbol{q}^{(t+1)} - \boldsymbol{q}^{(t)}\right) \frac{t_{\mathrm{FB}}}{\Delta t_{\mathrm{s}}} \tag{46}$$

$$\dot{\boldsymbol{\theta}}_{\mathrm{ref}} = \dot{\boldsymbol{\theta}}_{\mathrm{des}} + K_{\mathrm{p}}(\boldsymbol{\theta}_{\mathrm{des}} - \boldsymbol{\theta}_{\mathrm{cur}}) \tag{47}$$

where $\boldsymbol{q}_{\mathrm{des}} = (\boldsymbol{\theta}_{\mathrm{des}}, \dot{\boldsymbol{\theta}}_{\mathrm{des}})$ is the desired state, and $\boldsymbol{\theta}_{\mathrm{cur}}$ is the current angles of each joint. K_{p} is the feedback gain, and t_{FB} $(0 \leq t_{\mathrm{FB}} \leq \Delta t_{\mathrm{s}})$ is the time incremented by Δt_{FB}.

VI. Experiment

A. Experimental Setup

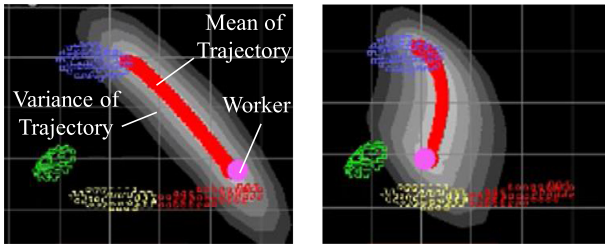
Fig. 7 shows the experimental environment, and Fig. 8 shows a top view of the experimental setup. It shows the arrangement of the robot and the sensor and the assumed working position and worker's path of motion. A laser range finder fixed outside the workspace is used to measure the worker's center position in two dimensions. In this experiment, the worker performs the following three tasks.

1) Tightening two bolts (Task 1).
2) Attaching three grommets and a brake tube (Task 2).
3) Attaching four plug holes (Task 3).

These parts were attached to the vehicle body above the worker's head. The worker started the first task from the starting position in Fig. 8 and moved to each working position in sequence, where he/she performed each task. After finishing Task 3, the worker moved to the goal position (the same position as the starting position), which complete one cycle of the experiment.

Before the worker starts each task, the robot delivers the parts and the tool to the worker. A robot that is always next to the worker can disturb not only a worker performing regular assembly tasks but also other workers who need to pass through the workspace. In this system, the robot moves only when the worker requires parts and tools. The robot aims to move so that the waste time for the worker to wait for the supply of the parts by the robot is as small as possible. In this experiment, this wasted time caused by the delay of the robot task is defined as the "waiting time." We define the waiting time in this paper as the time interval between the arrival of the worker at the next working position and the end of the robot's delivery movement. After the parts and tools are delivered, the robot moves back to its former position and remains there until the next task starts.

In this setup, the experiments were performed for five participants. Each participant first performed nine experiments with a regular motion pattern, and the worker's motion was learned incrementally. After the model learning was completed, one regular motion pattern and four irregular patterns were performed



(a) (b)

Fig. 9. Simulator for visualizing the prediction results and comparison of predicted trajectories for the regular case and irregular case. The colored distributions are the results of the working position model. The purple dot indicates the current position of the worker. The red line indicates the mean of the predicted trajectory. The white region indicates the variance of the predicted trajectory, which becomes dark gray as the existence probability of the worker decreases. (a) Regular case. (b) Irregular case.

TABLE I
SUMMARY OF PARAMETERS USED IN THIS EXPERIMENT FOR THE WORKER MOTION PREDICTOR

Parameters	Values
Measurement sampling time $\Delta t_{\rm s}$ (s)	0.03
Max prediction length of moving trajectory T_p	70
Total number of tasks for GMM $M_{\rm gmm}$	4
Initial variance for GMM in directions x and y $\sigma_{\rm ini}$ (m)	0.1
Threshold velocity used for task estimation $v_{\rm th}$ (m/s)	0.2
Auto regression order of GMR d	4
Upper significance probability used for percentile threshold α	0.05
Constant value for forgetting factor β	0.8

using the same work model. Here, we define the irregular pattern as the pattern in which the worker moves differently from the past movements but finally arrives at the target position. Irregular patterns were performed when the worker moved from the starting position to the position for Task 1. Consequently, each worker performed ten regular motion patterns and four irregular motion patterns. This was done for all of the participants and evaluated for a total of 70 movements (50 regular motion patterns and 20 irregular motion patterns). The specific irregular motion pattern was determined individually by each participant.

Fig. 9 shows a simulator used for these experiments to visualize the prediction results. It shows the likelihood of each working position (colored distributions in the figure) and the predicted trajectory (red line in the figure) with its covariance (white region in the figure). Fig. 9 also shows a comparison of predicted trajectories calculated in the regular case (left image) and the irregular case (irregular case). Compared to the regular case, there is a tendency that the variance of the predicted trajectory becomes large in the irregular case.

The parameters of the worker motion predictor used in the experiments are summarized in Table I. The length of the predicted trajectory $T_p = 70$ is determined from the actual moving time of the worker. Because the measurement sampling time is $\Delta t_{\rm s} = 0.03$, this corresponds to about 2 s. The initial covariance $\sigma_{\rm ini}$ of the GMM is determined empirically by considering the variance in the worker position at the working position and the sensor resolution. Considering a human speed and sensor noise, we set the velocity threshold $v_{\rm th}$ to be 0.2 m/s. The GMR order d is adjusted by the prior simulation while taking the actual prediction results into account. In this experiment, we use $d = 4$. If d is too small, the accuracy of the prediction for sudden movements becomes poor, whereas if d is too large, the delay of the predicted trajectory will become large. We use the upper significance probability $\alpha = 0.05$, which are generally used as outlier detection in the statistical analysis. This means the fifth percentile is used for threshold $a_{{\rm th},m}(\alpha)$. For the task estimation using the conditions (3) and (42), threshold becomes $a_{{\rm th},2}(0.05) \simeq 5.99$. For the unit addition and deletion operation using the online learning Algorithm 2, threshold becomes $a_{{\rm th},2\times4}(0.05) \simeq 15.51$. For the constant value β, we use $\beta = 0.8$ that should satisfy $0.5 < \beta \leq 1.0$ to ensure the convergence of online learning. The larger value of β leads to the larger forgetting rate for old parameters.

TABLE II
SUMMARY OF PARAMETERS USED IN THIS EXPERIMENT FOR THE ONLINE TRAJECTORY GENERATOR AND ROBOT CONTROL SYSTEM

Parameters	Values
Control sampling time $\Delta t_{\rm FB}$ (s)	0.001
Weight matrix R $(R_{11}, R_{22}, R_{33}, R_{44})$	$(400, 400, 30, 30)$
Weight coefficients (r_1, r_2)	$(1000, 1000)$
Weight coefficient w	100
Max angular velocity $(\dot{\theta}_{1,\max}, \dot{\theta}_{2,\max})(rad/s)$	(π, π)
Max angular acceleration $(\ddot{\theta}_{1,\max}, \ddot{\theta}_{2,\max})(rad/s^2)$	(π, π)
Small constant for determining convergence ϵ	0.001

The parameters used for the online trajectory generator and robot control system are summarized in Table II. As a control sampling time $\Delta t_{\rm FB}$, we use $\Delta t_{\rm FB} = 0.001$ in order to make the robot follow the desired trajectory stably. The weighting coefficients r_1 and r_2 for considering the constraint conditions of angular velocities are set very large so as not to exceed the constraint condition. As for the maximum angular velocities $\dot{\theta}_{\max,1}$ and $\dot{\theta}_{\max,2}$ and the maximum angular accelerations $\ddot{\theta}_{\max,1}$ and $\ddot{\theta}_{\max,2}$, small values are set since it affects the smoothness of the generated trajectory. In the target case, since the robot needs to approach the worker in order to supply the parts and the tool, the weighting coefficients R and w affect a tradeoff relationship between the work-time efficiency and worker's safety. In this case, the coefficient R affects the speed of convergence to the target position, whereas the coefficient w affects the distance between the worker and the robot's endpoint when avoiding the worker. Fig. 10 shows the results of the simulations done to examine the effect of the coefficient w. In the simulation, R is fixed at the values shown in Table II. The results confirm that a larger value of w leads to a larger minimum distance between the worker and the robot's endpoint in the irregular case. However, a larger value of w also leads to a longer waiting time in the regular case. In this experiment, we use $w = 100$ to prioritize the worker's safety. In this case, the robot's endpoint should remain a constant 0.3 m from the worker.

B. Experimental Result

1) Regular Cases: Fig. 11 shows an example of the experiment when the worker moves in the regular motion pattern. This example shows the result of the tenth trial of participant A. The images on the left side of each scene show an actual work in the real space and the images on the right side show

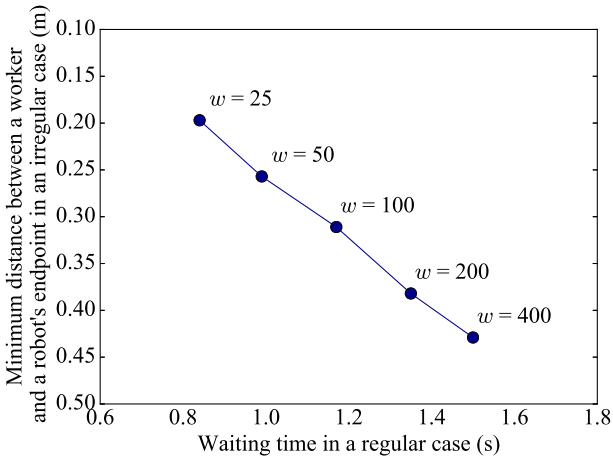


Fig. 10. Weight coefficients w for waiting time versus the minimum distance between a worker and the robot's endpoint calculated in the simulation.

the results of the prediction of the worker's motion. The robot's optimal trajectory is calculated while the system predicts the behavior of the worker. The results confirm that the worker can carry out the assembly work while the robot delivers the parts and tools to the worker. For other subjects, we also confirmed that they could accomplish the assembly work while receiving parts from the robot.

For regular cases, we evaluated the experimental results based on the waiting time caused by a delay in the delivery of parts to the worker. Figs. 12–14 show the waiting time for ten trials for each participant executing regular motion patterns, including the results when moving from Start to Task 1, from Task 1 to Task 2, and from Task 2 to Task 3. Since the waiting time is greatly affected by the change of the way how worker moves, its variation between trials becomes large. However, the results confirm that the waiting time tends to be reduced for all participants as the trial is repeated. Because the worker moves a long distance in going from Start to Task 1 and from Task 1 to Task 2, the waiting times are greatly reduced. However, when moving from Task 2 to Task 3, the decrease in the waiting time is not so significant because the worker moves a relatively small distance (about 0.4 m). In this system, the worker model is constructed using only data for the worker's center position. Therefore, the effect of the predicted trajectory becomes small for the situation with less movement. We consider that motionless work could also be dealt with by incorporating the direction of the worker and the movement of the worker's arm into the worker model.

2) Irregular Cases: Fig. 15 shows a typical experiment dealing with an irregular motion pattern as performed by the participant B for case 3. In this experiment, once the worker left the working position, he approached the robot's endpoint before the robot reached the target position. In this case, the robot moved away from the worker, and after the worker reached the target position, the robot approached the target position and completed its task as soon as possible. At the time shown in Fig. 15, the robot is approaching the target position from the direction of the smallest variance in the worker's predicted trajectory. This strategy contributes to minimizing the risk of collision with the worker.

For irregular motion patterns, the evaluation is based on the distance between the worker and the robot's endpoint during delivery. In following results, we also show the simulation results of the planned trajectory assuming that the variance of the predicted worker's position in (13) is constant and small in order to compare with the proposed planning method. Considering the variance of the worker's movement in each trial and the predicted variance in regular cases, we set the constant variance $\mathbf{\Sigma}_{\mathrm{const}}$ as $\mathbf{\Sigma}_{\mathrm{const},11} = \mathbf{\Sigma}_{\mathrm{const},22} = 0.1$ m. For the mean vector of the worker's position, we used the result calculated by the worker motion predictor. The planned robot's movement considering the preset constant variance is shown in the attached video.

Fig. 16 shows the paths of the robot's endpoint and of the worker. It shows three trajectories: planned in the regular case (black line), planned in the irregular case considering the preset constant variance (blue line), and planned in the irregular case considering the predicted variance (green line). It can be confirmed that the planning method considering the increase of uncertainty due to the irregular motion planned the robot's trajectory, which avoids the worker by taking a larger distance than other methods. Fig. 17 shows the time-series data of the distance between the worker and the robot's endpoint for the case 3 of the participant B. It is confirmed that the worker and the endpoint maintain a certain distance over 0.3 m while the delivery task is being accomplished. However, the planning method considering the preset constant variance planned the trajectory, which moved closer to the worker than the method considering the predicted variance.

Fig. 18 shows the results for the distance of the closest approach when each worker moves in an irregular pattern. The simulation results of the trajectory considering the constant variance confirm that the robot approached within 0.3 m of the worker in many cases. On the other hand, the results of the trajectory considering the predicted variance confirm that the robot remains about 0.3 m from the worker. When the worker moved in an irregular way, the proposed method greatly reduced the risk of collision with the worker by considering the increase in uncertainty of the predicted worker movement. For the two cases, the minimum distance is slightly less than 0.3 m because collision avoidance is treated as a soft constraint not a hard constraint in the optimization problem. In these experiments, no collision between the worker and the robot occurred.

3) Discussion: The experiments described previously confirm that the system set with the same parameters can deal with both regular and irregular motion patterns of several workers. In the regular case, the proposed system can reduce the waiting time of the worker and improve the work-time efficiency. In the irregular case, the proposed system can complete the original robot's task while avoiding contact with the worker, thereby, ensuring the worker's safety. The effect of these two functions is in a tradeoff relationship, which results from the weighting coefficients w in (29) and which must be determined beforehand. Tuning the parameters enables a system administrator to arbitrarily select the priority between the scheduled delivery time or minimizing of the risk of the collision with the worker. Therefore, we consider that the system parameters must be converted into a form that facilitates their tuning. Specifically, if the

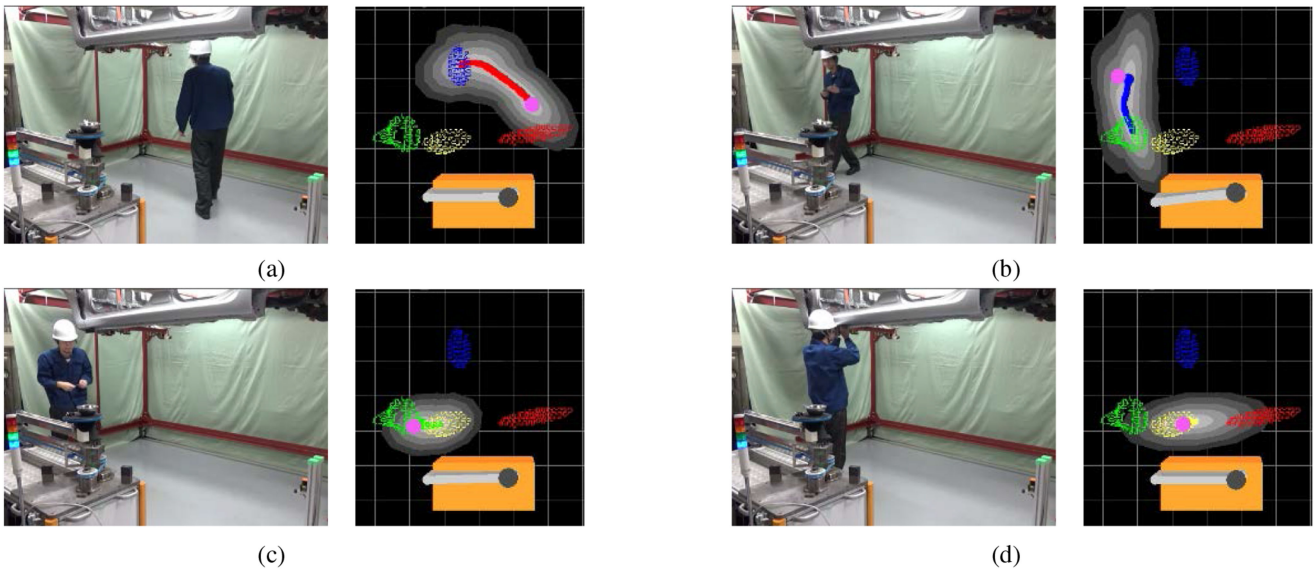


Fig. 11. Assembly work with a collaborative robot in the case when the worker executes a regular motion pattern (trial 10 of participant A). The left images show the real environment, and the right images show the motion predicted by the system. The full experiment is shown in the attached video. (a) Moves to Task 1. (b) Moves to Task 2. (c) Moves to Task 3. (d) Assembles parts for Task 3.

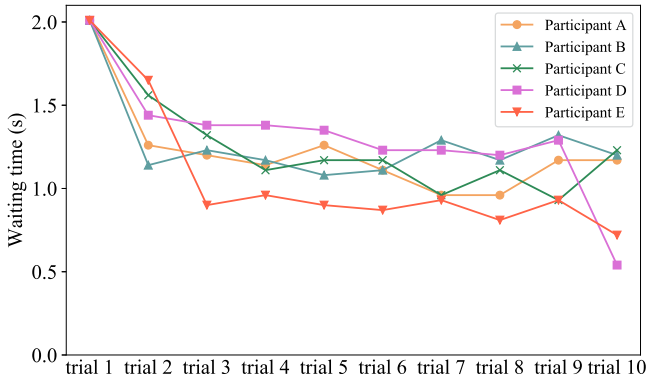


Fig. 12. Waiting time for each participant when moving from Start to Task 1.

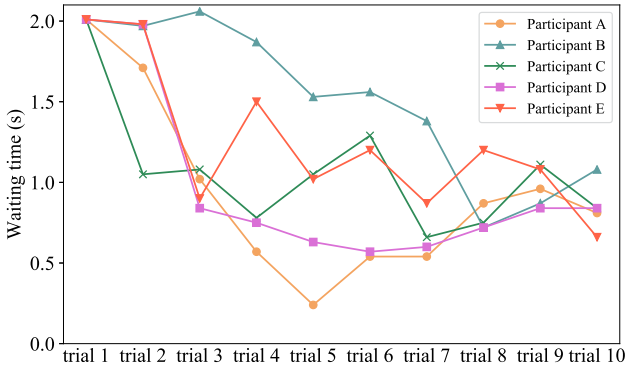


Fig. 13. Waiting time for each participant when moving from Task 1 to Task 2.

parameters could be converted to a tolerable waiting time or the minimum distance between the worker and the robot, it would be possible for humans to set them intuitively, which would facilitate the use of the system.

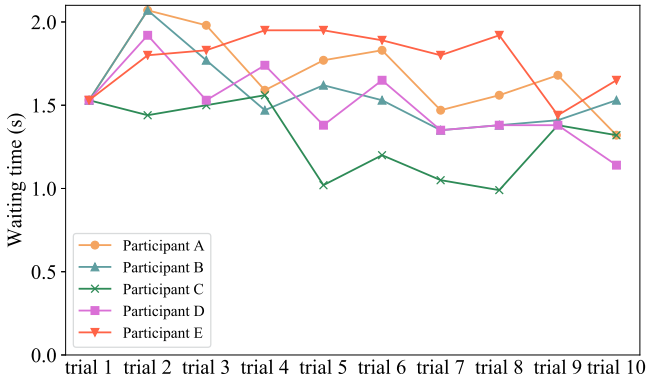


Fig. 14. Waiting time for each participant when moving from Task 2 to Task 3.

In addition, the results confirm that, with a given set of system parameters, the proposed system can deal with several workers. In particular, robot's trajectories that do not collide with workers can be generated even when each worker executes various irregular motion patterns. However, this experiment was done under the constraint that the worker eventually reaches the target position after the irregular motion. If an abnormal motion pattern occurs (e.g., the worker departs from the workspace), the robot's trajectory is not guaranteed to converge to the target position. Moreover, even when the worker executes a regular motion pattern, the system may not operate properly because of system errors, such as tracking failure by the sensor and prediction failure by the system. Therefore, we considered it essential for the system applied to the actual environment to be able to calculate the degree of abnormality in the workspace. When the degree of abnormality is extremely large, the system must stop the robot's task in a safe manner.

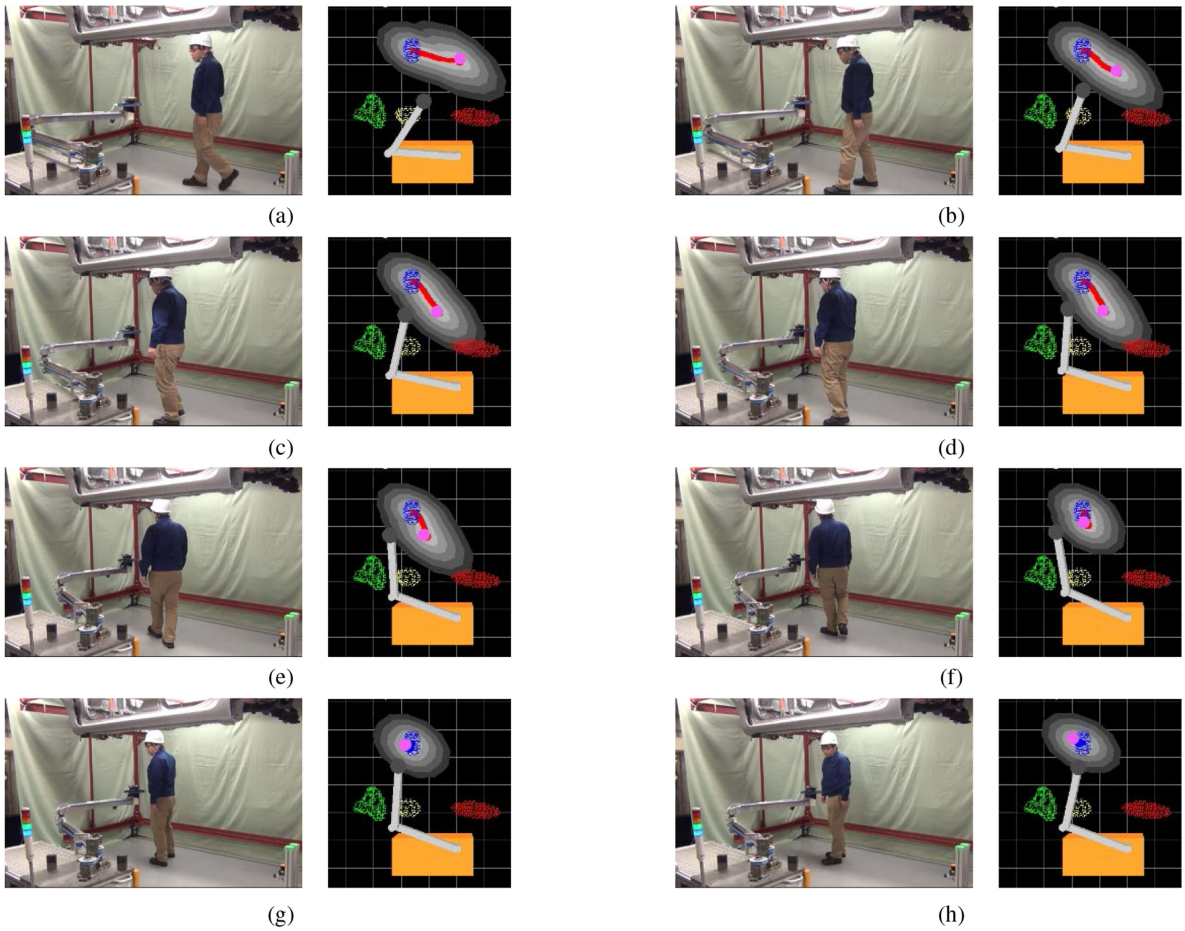


Fig. 15. Assembly work with a collaborative robot in the case where the worker executes an irregular pattern (case 3 of participant B). The left images show the real environment, and the right images show the predicted motion. The full experiment is shown in the attached video. (a) 4.0 s. (b) 4.5 s. (c) 5.0 s. (d) 5.5 s. (e) 6.0 s. (f) 6.5 s. (g) 7.0 s. (h) 7.5 s.

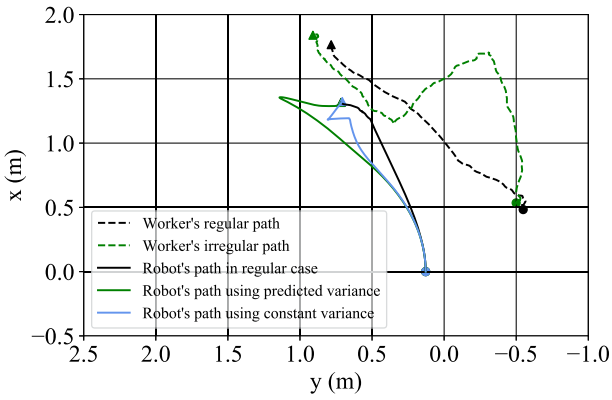


Fig. 16. Paths of robot endpoint and of the worker when parts are being delivered. The black line indicates the paths in the regular case. (trial 10 of participant B). The green line indicates the paths in the irregular case (case 3 of participant B). The blue line indicates the planned trajectory assuming that the variance of worker's movement is constant and small. The circles indicate the start point of the path and the triangles indicate the goal point of the path.

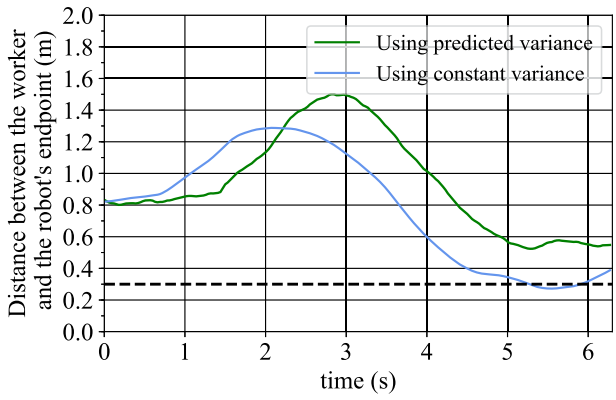


Fig. 17. Distance between robot endpoint and worker when parts are being delivered in the case of irregular motion (case 3 of participant B). The dashed line indicates the minimum robot-worker separation of 0.3 m, which is assumed to be maintained to ensure the worker's safety in the preliminary simulation.

Finally, the guarantee of the collision avoidance should be improved. In the optimization problem, the collision avoidance is imposed with soft constraint so that the robot can get close to the worker for supplying the parts and tools. However, the worker's safety should be given the highest priority and collision avoidance should be imposed as a hard constraint. We consider there is room for the improvement in the handling of a tradeoff relationship between the worker's safety and the

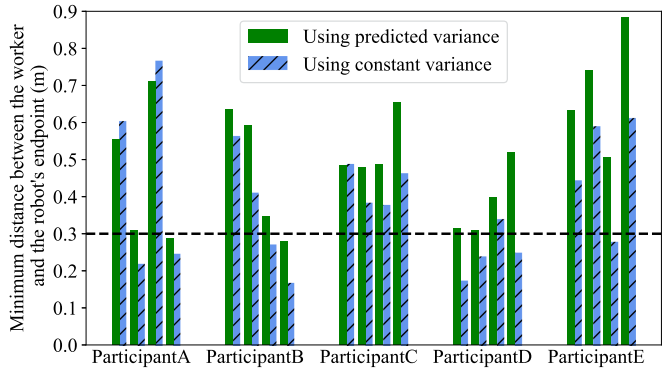


Fig. 18. Minimum distances between the robot's endpoint and worker for each participant. The dashed line indicates the minimum robot–worker separation of 0.3 m, which is assumed to be maintained to ensure the worker's safety in the preliminary simulation.

work-time efficiency. It is necessary to grasp the target problem as a multiobjective optimization problem with a high priority on the worker's safety.

VII. Conclusion

In this paper, we proposed the adaptive motion planning system for a collaborative robot to operate in assembly processes. The online trajectory generator based on the receding horizon scheme was installed in the proposed system to exploit the predicted worker motion. Because of the online trajectory generator, which generates the robot's trajectory based on a probabilistic prediction of the worker's motion, the proposed system can deal simultaneously with the cases of regular and irregular motion. In the case of regular motion, the proposed system reduces the waiting time of the worker by explicitly considering the temporal requirements in the trajectory optimization. In addition, the collaborative robot can complete its original task while avoiding contact with the worker who moves irregularly in the workspace by considering the prediction uncertainty in the optimization problem. To experimentally evaluate the proposed system, it was applied to an assembly scenario with the two-link planar manipulator. The results of the experiments involving several workers confirmed that the proposed system can simultaneously enhance the work-time efficiency and the worker's safety.

In future work, we are planning to extend the functionality of the proposed system so that it can be installed in a wide variety of actual processes. This would require a system to detect abnormal states and to evaluate the degree of the abnormality in a given process to determine whether the robot system should be stopped. The implementation of such a system would completely ensure the safety of workers against unforeseen circumstances. In addition, the proposed system should be applied to and evaluated with other assembly processes. Thus, the combination of a system to predict worker motion from data of past worker motion and a trajectory generation system with a robot with multiple degrees of freedom provides a system that can flexibly deal with various assembly processes.

References

[1] M. Bélanger-Barrette, *Collaborative Robot eBook*. Levis, QC, Canada: ROBOTIQ, Oct. 2015.
[2] J. Kinugawa, A. Kanazawa, S. Arai, and K. Kosuge, "Adaptive task scheduling for an assembly task coworker robot based on incremental learning of human motion patterns," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 856–863, Apr. 2017.
[3] J. Fryman and B. Matthias, "Safety of industrial robots: From conventional to collaborative applications," in *Proc. German Conf. Robot.*, 2012, pp. 1–5.
[4] N. Pedrocchi, F. Vicentini, M. Matteo, and L. M. Tosatti, "Safe human-robot cooperation in an industrial environment," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 1, pp. 1–13, 2013.
[5] C. Liu and M. Tomizuka, "Algorithmic safety measures for intelligent industrial co-robots," in *Proc. Int. Conf. Robot. Autom.*, 2016, pp. 3095–3102.
[6] P. A. Lasota, G. F. Rossano, and J. A. Shah, "Toward safe close-proximity human-robot interaction with standard industrial robots," in *Proc. Int. Conf. Autom. Sci. Eng.*, 2014, pp. 339–344.
[7] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, and B. Matthias, "Safety in human-robot collaborative manufacturing environments: Metrics and control," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 882–893, Apr. 2016.
[8] R. Wilcox, S. Nikolaidis, and J. Shah, "Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing," in *Proc. Robot. Sci. Syst.*, 2012, pp. 441–448.
[9] K. P. Hawkins, N. Vo, S. Bansal, and A. F. Bobick, "Probabilistic human action prediction and wait-sensitive planning for responsive human-robot collaboration," in *Proc. Int. Conf. Humanoid Robots*, 2013, pp. 499–506.
[10] K. Baizid, A. Yousnadj, A. Meddahi, R. Chellali, and J. Iqbal, "Time scheduling and optimization of industrial robotized tasks based on genetic algorithms," *Robot. Comput.-Integr. Manuf.*, vol. 34, pp. 140–150, 2015.
[11] R. Hayne, R. Luo, and D. Berenson, "Considering avoidance and consistency in motion planning for human-robot manipulation in a shared workspace," in *Proc. Int. Conf. Robot. Autom.*, 2016, pp. 3948–3954.
[12] N. M. Ceriani, A. M. Zanchettin, P. Rocco, A. Stolt, and A. Robertsson, "Reactive task adaptation based on hierarchical constraints classification for safe industrial robots," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 6, pp. 2935–2949, Dec. 2015.
[13] D. Vasquez, T. Fraichard, and C. Laugier, "Incremental learning of statistical motion patterns with growing hidden Markov models," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 403–416, Sep. 2009.
[14] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2287–2301, Nov. 2011.
[15] F. Dawood, C. K. Loo, and W. H. Chin, "Incremental on-line learning of human motion using gaussian adaptive resonance hidden Markov model," in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–7.
[16] S. Calinon, A. Pistillo, and D. G. Caldwell, "Encoding the time and space constraints of a task in explicit-duration hidden Markov model," in *Proc. Int. Conf. Intell. Robots Syst.*, 2011, pp. 3413–3418.
[17] D. W. Park, J. Kwon, and K. M. Lee, "Robust visual tracking using autoregressive hidden Markov model," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1964–1971.
[18] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *Proc. Int. Symp. Robot. Res.*, 2003, pp. 561–572.
[19] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Proc. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2616–2624.
[20] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *Proc. Int. Conf. Humanoid Robots*, 2012, pp. 323–329.
[21] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with gaussian mixture models," in *Proc. Intell. Veh. Symp.*, 2012, pp. 141–146.
[22] J. Wang, A. Hertzmann, and D. J. Fleet, "Gaussian process dynamical models," in *Proc. Conf. Neural Inf. Process. Syst.*, 2006, pp. 1441–1448.
[23] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," *Auton. Robots*, vol. 35, no. 1, pp. 51–76, 2013.

[24] K. Kim, D. Lee, and I. Essa, "Gaussian process regression flow for analysis of motion trajectories," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 1164–1171.
[25] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. Auton. Robot Veh.*, 1986, pp. 396–404.
[26] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *Proc. Int. Conf. Robot. Autom.*, 1993, pp. 802–807.
[27] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.
[28] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Computer Science Dept., Iowa State Univ., Ames, IA, USA, Tech. Rep. 98-11, 1998.
[29] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Trans. Comput.*, vol. C-32, no. 2, pp. 108–120, Feb. 1983.
[30] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. Int. Conf. Robot. Autom.*, 2000, pp. 995–1001.
[31] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.
[32] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa,"CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. Int. Conf. Robot. Autom.*, 2009, pp. 489–494.
[33] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. Int. Conf. Robot. Autom.*, 2011, pp. 4569–4574.
[34] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Proc. Int. Conf. Automated Planning Scheduling*, 2012, pp. 207–215.
[35] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," *Robot., Sci. Syst.*, vol. 9, no. 1, pp. 1–10, 2013.
[36] J. J. Kim and J. J. Lee, "Trajectory optimization with particle swarm optimization for manipulator motion planning," *IEEE Trans. Ind. Inform.*, vol. 11, no. 3, pp. 620–631, 2015.
[37] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *Proc. Int. Conf. Robot. Autom.*, 2014, pp. 3339–3344.
[38] J. R. Hernández, D. Lee, and S. Hirche, "Risk-sensitive optimal feedback control for haptic assistance," in *Proc. Int. Conf. Robot. Autom.*, 2012, pp. 1025–1031.
[39] G. Tonietti, R. Schiavi, and A. Bicchi, "Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction," in *Proc. Int Conf. Robot. Autom.*, 2005, pp. 526–531.
[40] N. E. Du Toit and J. W. Burdick, "Robot motion planning in dynamic, uncertain environments," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 101–115, Feb. 2012.
[41] T. Ohtsuka, "A continuation/GMRES method for fast computation of nonlinear receding horizon control," *Automatica*, vol. 40, no. 4 , pp. 563–574, 2004.
[42] J. Kinugawa, Y. Sugahara, and K. Kosuge, "Co-worker robot," *Acta Polytechnica Hungarica*, vol. 13, no. 1, pp. 209–221, 2016.

Akira Kanazawa (M'18) received the M.S. degree in bioengineering and robotics from Tohoku University, Sendai, Japan, in 2017.

He is a Researcher with Hitachi Ltd., Hitachinakashi, Japan. His research interests include human motion modeling and robot motion planning.

Mr. Kanazawa is a member of the IEEE Robotics and Automation Society.

Jun Kinugawa (M'13) received the Ph.D. degree in bioengineering and robotics from Tohoku University, Sendai, Japan, in 2011.

He is an Assistant Professor with the Department of Robotics, Tohoku University. His research interests include assembly task partner robot, robot hand fabricated via the shape deposition manufacturing scheme, and power assist system.

Dr. Kinugawa is a member of the IEEE Robotics and Automation Society.

Kazuhiro Kosuge (F'06) received the B.S., M.S., and Ph.D. degrees in control engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1978, 1980, and 1988 respectively.

He is currently a Professor with the Department of Robotics, Tohoku University, Sendai, Japan. From 1980 to 1982, he was a Research Staff with the Production Engineering Department, Nippon Denso Company Ltd. (DENSO Company Ltd.). From 1982 to 1990, he was a Research Associate with the Department of Control Engineering, Tokyo Institute of Technology. From 1990 to 1995, he was an Associate Professor with Nagoya University. Since 1995, he has been with Tohoku University.

Prof. Kosuge was the recipient of the JSME Awards for the best papers from the Japan Society of Mechanical Engineers in 2002 and 2005, the RSJ Award for the best papers from the Robotics Society of Japan in 2005. He is a Fellow of the Japan Society of Mechanical Engineers, the SICE, the Robotics Society of Japan, and the Society of Automotive Engineers of Japan. He was the President of the IEEE Robotics and Automation Society for 2010–2011 and the IEEE Division X Director for 2015–2016. He is a member of the IEEE-Eta Kappa Nu and the IEEE Vice President for Technical Activities for 2019.